Title: The adaptation of the harmony search algorithm to the ATSP with the evaluation of the influence of the pitch adjustment place on the quality of results

Author: Urszula Boryczka, Krzysztof Szwarc

Citation style: Boryczka Urszula, Szwarc Krzysztof. (2018). The adaptation of the harmony search algorithm to the ATSP with the evaluation of the influence of the pitch adjustment place on the quality of results. “Journal of Information and Telecommunication”, doi 10.1080/24751839.2018.1503149

JOURNAL OF INFORMATION AND TELECOMMUNICATION
https://doi.org/10.1080/24751839.2018.1503149

The adaptation of the harmony search algorithm to the ATSP with the evaluation of the influence of the pitch adjustment place on the quality of results

Urszula Boryczka and Krzysztof Szwarc

Institute of Computer Science, University of Silesia in Katowice, Sosnowiec, Poland

ABSTRACT

The paper is an extended version of the conference article, which presents a modification of the Harmony Search algorithm, adapted to the effective resolution of the asymmetric case of the Traveling Salesman Problem. The efficacy of the proposed approach was measured with benchmarking tests and in a comparative study based on the results obtained with the Nearest Neighbor Algorithm, Greedy Local Search and Hill Climbing. The discussion also embraced the study of the convergence of the proposed algorithm and the analysis of the impact of the pitch adjustment place on the quality of the solutions.

ARTICLE HISTORY
Received 24 April 2018
Accepted 19 July 2018

KEYWORDS
Harmony search; asymmetric traveling salesman problem; metaheuristics; pitch adjustment place

1. Introduction

The Harmony Search (HS) algorithm is a promising metaheuristic used to solve a variety of optimization problems (it has been successfully used in the design of steel frames by Degertekin, 2008, the routing optimization in 4PL with time windows by Bo, Huang, lp, & Wang, 2009, the Internet routing by Forsati, Haghighat, & Mahdavi, 2008, the optimization of container storage in a harbor area by Ayachi, Kammarti, Ksouri, & Borne, 2010 and in the flexible job shop scheduling problem with multiple objectives by Gao et al., 2016). Its results are usually characterized with the favorable value of the objective function, while at the same time they are achieved in a relatively short time. The nature of the algorithm, which determines the preferential use of the method for continuous optimization problems, requires the application of sophisticated approaches that allow for its adaptation to other ways of representing a number of important issues in business practice.

The Traveling Salesman Problem (TSP) is a classical combinatorial optimization problem that involves finding the shortest Hamiltonian cycle in a complete weighted graph. Its popularity stems from the fact that it belongs to the class of *NP*-hard problems and that it can model a variety of utilitarian issues – its asymmetric variant (characterized by the possibility of varying weights of edges connecting the same nodes), representing line

CONTACT Krzysztof Szwarc krzysztof.szwarc@us.edu.pl Institute of Computer Science, University of Silesia in Katowice, ul. Będzińska 39, 41-200 Sosnowiec, Poland

infrastructure located in urban areas, has become the basis for many logistical problems (it models, for example, the process of planning the mobile collection of waste electrical and electronic equipment, which has been described by Mrówczyńska & Nowakowski, 2015; Nowakowski, Szwarc, & Boryczka, 2018, and the process of transport activities related to the acquisition of municipal waste, described by Płaczek & Szołtysek, 2008; Syberfeldt, Rogstrom, & Geertsen, 2015).

Taking into account the relatively good research results concerning the use of the Harmony Search algorithm to solve many practical problems, a number of controversies over HS (according to Weyland, 2010 it is assumed that the method is non-innovative and it is only a special case of Evolution Strategies), and the utilitarian importance of the asymmetric variant of the Traveling Salesman Problem, we chose to conduct a study aimed at adapting the cited metaheuristics to the combinatorial optimization problem. The proposed topic was also discussed by Antosiewicz, Koloch, and Kamiński (2013), but the results presented by them show the ineffectiveness of the method adapted to TSP, implying the need for an innovative approach to the design of the algorithm facilitating the process of planning the traveling salesman's route.

The paper consists of the following parts: the first part introducing the topic, the second part presenting the Harmony Search algorithm, the third part formulating the asymmetric problem of the traveling salesman, the fourth part proposing the approach to adapt the metaheuristic, the fifth part describing the methodology of the study, the sixth part discussing the results, and the seventh part concentrating on the conclusions and planned work.

The paper is an extended version of an article authored by Boryczka and Szwarc (2018), additionally enriched with the study of the convergence of the proposed algorithm and the analysis of the impact of the pitch adjustment place on the quality of the solutions in our approach to the HS design. In Section 1 we have added information about new applications of the HS and additional references for ATSP applications. We have changed the formulation of the ATSP in Section 3 and added the last paragraph and the Figure 3 in Section 5. Additionally, in the Results we have addded the last four paragraphs and Tables 6–10. Article was also extended by the last paragraph in Section 7.

2. Harmony search algorithm

The Harmony Search technique, proposed in the paper of Geem, Kim, and Loganathan (2001), is based on the similarity of the jazz improvization process to the search for a global optimum by algorithmic methods. It assumes the existence of a *HM* structure (referred to as harmony memory), which stores *HMS* harmonies (Panchal, 2009 said that usually from 4 to 10), consisting of a specified number of pitches (representing the values of the decision variables of a given result). Each *HM* element is interpreted as a complete solution to a problem whose objective function value is determined based on its components.

The initial harmony memory content is generated randomly and later aligned according to the appropriate objective function values (adapted to the analyzed problem), which describe individual harmonies (so that the first result in *HM* is the most favorable). These steps initiate the iterative creation of successive solutions.

The Harmony Search pseudocode

```
function HS(HMS, HMCR, PAR, IT, bw)
    iterations = 0
    for i = 0; i < HMS; i++ do
        HM[i]=stochastically generate feasible solution
    end for
    Sort HM
    while iterations < IT do
        for i = 0; i < n; i++ do                    ▷ n - number of pitches
            Choose random r ∈ (0, 1)
            if r < HMCR then
                H[i]=choose randomly available pitch on position i in HM
                Choose random k ∈ (0, 1)
                if k < PAR then
                    α = bw· random ∈ (−1, 1)        ▷ bw - range of changes
                    H[i]=H[i] + α
                end if
            else
                H[i]=choose randomly available pitch
            end if
        end for
        if f(H) is better than f(HM[HMS − 1]) then
            HM[HMS − 1] = H
            Sort HM
        end if
        iterations++
    end while
    return HM[0]
end function
```

Figure 1. The Harmony Search pseudocode based on Geem et al. (2001) and Hetmaniok, Jama, Słota, and Zielonka (2011).

The procedure for creating a new solution uses the knowledge accumulated in *HM* and is based on the analogy to the improvization of harmony in music. The development of a solution involves an iterative selection of the next pitch, according to two parameters – *HMCR* (a harmony memory considering rate; as Ayachi et al., 2010 noticed its value is usually within the range of 0.7 to 0.99) and *PAR* (a pitch adjustment rate; as Ayachi et al., 2010 said it is often from 0.1 to 0.5). Based on the probability of *HMCR*, the pitch *i* is selected, using the values in the *i* position in harmonies belonging to *HM* (otherwise the value is generated randomly). Creating a solution based on the *HM* component, the pitch can be modified with a defined probability of *PAR* (the change in value is based on the *bw* parameter, the value of which depends on the representation of a problem).

When the next solution is generated, a comparison of its objective function value with the relevant parameter describing the *HM* component in the last position is made. When a more favorable result is identified, it replaces the worst result in harmony memory and *HM* elements are rearranged.

The Harmony Search pseudocode for ATSP

```
1:  function HS(HMS, HMCR, PAR, IT, R, first city)
2:      iterations = 0
3:      iterationsFromTheLastReplacement = 0
4:      for i = 0; i < HMS; i++ do
5:          HM[i]=stochastically generate feasible solution
6:      end for
7:      Sort HM
8:      while iterations < IT do
9:          H[0]=first city
10:         for i = 1; i < n; i++ do                    ▷ n - number of cities
11:             Choose random r ∈ (0, 1)
12:             if r < HMCR then
13:                 list=generate list containing vertices occurring after H[i−1] in HM
14:                 if list.length > 0 then
15:                     H[i]=choose element ∈ list according to the roulette wheel
16:                 else
17:                     H[i]=choose randomly available city ∉ H
18:                 end if
19:                 Choose random k ∈ (0, 1)
20:                 if k < PAR then
21:                     H[i]=find nearest and available city from H[i − 1]
22:                 end if
23:             else
24:                 H[i]=choose randomly available city ∉ H
25:             end if
26:         end for
27:         if f(H) is better than f(HM[HMS − 1]) then
28:             HM[HMS − 1] = H
29:             Sort HM
30:             iterationsFromTheLastReplacement = 0
31:         else
32:             iterationsFromTheLastReplacement++
33:         end if
34:         if iterationsFromTheLastReplacement = R then
35:             for i = 1; i < HMS; i++ do
36:                 HM[i]=stochastically generate feasible solution
37:             end for
38:             Sort HM
39:             iterationsFromTheLastReplacement = 0
40:         end if
41:         iterations++
42:     end while
43:     return HM[0]
44: end function
```

Figure 2. The Harmony Search pseudocode for ATSP.

The procedure for generating a new solution is performed by *IT* iterations, and then the best result (in the first position in harmony memory) is returned. The pseudocode of the method is shown in Figure 1.

Table 1. Characteristics of tests based on Osaba, Diaz, Onieva, Carballedo and Perallos (2014).

No.	Name	No. of vertices	Optimum
1	br17	17	39
2	ftv33	34	1286
3	ftv35	36	1473
4	ftv38	39	1530
5	p43	43	5620
6	ftv44	45	1613
7	ftv47	48	1776
8	ry48p	48	14422
9	ft53	53	6905
10	ftv55	56	1608
11	ftv64	65	1839
12	ft70	70	38673
13	ftv70	71	1950
14	kro124p	100	36230
15	ftv170	171	2755
16	rbg323	323	1326
17	rbg358	358	1163
18	rbg403	403	2465
19	rbg443	443	2720

3. The formulation of the asymmetric traveling salesman problem

Based on the paper of Öncan, Altınel, and Laporte (2009), the following TSP formulation was adapted: for a directed graph $G = (V, A)$, with weighted arcs represented as c_{ij} (where $i, j \in \{1, 2, \ldots, n\}$), a route (a directed cycle comprising all n cities) of minimal length is sought. The asymmetric variant of the Traveling Salesman Problem (ATSP) is characterized with the possibility of the occurrence of inequality $c_{ij} \neq c_{ji}$.

A decisive variable x_{ij} representing the edge between vertices i and j in the solution found adapts the following values:

$$x_{ij} = \begin{cases} 1 & \text{when the edge } (i, j) \text{ is part of the route constructed,} \\ 0 & \text{otherwise.} \end{cases} \tag{1}$$

The objective function was formulated in the following way:

$$\sum_{i=1}^{n} \sum_{j=1}^{n} c_{ij} x_{ij} \rightarrow \min . \tag{2}$$

The constraints ensuring exactly one visit of the traveling salesman to every city were presented in the following way:

$$\sum_{i=1}^{n} x_{ij} = 1, \quad j = 1, \ldots, n, \quad \sum_{j=1}^{n} x_{ij} = 1, \quad i = 1, \ldots, n. \tag{3}$$

Such a formulation of the Traveling Salesman Problem could result in the occurrence of solutions representing separate cycles instead of 1 cycle, so it is necessary to introduce additional constraints (MTZ):

$$1 \leq u_i \leq n - 1, \quad u_i - u_j + (n - 1) x_{ij} \leq n - 2, \quad i, j = 2, \ldots, n. \tag{4}$$

Table 2. The impact of parameter values on the average error obtained after 1 min.

	Average error																			
	R						*HMS*				*HMCR*					*PAR*				
Test name	lack	250	500	750	1000	1250	3	5	7	10	0.95	0.97	0.98	0.99	1	0.2	0.24	0.25	0.26	0.3
p43	0.07	0.04	0.05	0.05	0.05	0.05	0.05	0.05	0.05	0.05	0.05	0.06	0.05	0.05	0.05	0.05	0.05	0.05	0.05	0.05
ry48p	3.28	1.74	1.66	2.17	1.63	1.64	1.74	1.63	1.96	2.29	1.63	1.88	1.35	1.8	3.99	2.14	1.68	1.35	1.89	1.36
ft70	5.69	6.03	5.76	5.45	5.49	5.58	5.73	5.49	5.65	5.59	5.49	5.07	4.75	4.7	4.41	4.74	4.89	4.75	4.75	5.09
Average	3.01	2.6	2.49	2.56	**2.39**	2.42	2.51	**2.39**	2.55	2.65	2.39	2.34	**2.05**	2.18	2.82	2.31	2.21	**2.05**	2.23	2.17

Table 3. The parameters of the laptop used to conduct the study.

No.	Parameter	Value
1	Processor	Intel Core i7-4720HQ (4 cores, from 2.60 GHz to 3.60 GHz, 6 MB cache)
2	RAM	16 GB (SO-DIMM DDR3, 1600 MHz)
3	Hard Drive	1000 GB SATA 5400 rev. Express Cache 8 GB
4	Operating system	Windows 7 Professional N Service Pack 1 64-bit

4. The proposal of the approach to HS design

According to the proposed approach to the design of the HS method (tailored to solve the Asymmetric Traveling Salesman Problem), each pitch is represented by integers corresponding to the number of the individual cities visited by the sales agent. The order of their occurrence – in harmony representing the complete route – indicates the sequence of a journey.

Taking into account the nature of the optimization problem under study, the position occupied by a given city in harmony is deemed irrelevant. It is necessary, however, to consider the sequence of vertices, by selecting the next pitch value based on the generated list of available nodes, occurring in the solutions recorded directly after the last city that belongs to the constructed result. Based on the structure created, the city is selected according to the roulette wheel method (the probability of acceptance of a given item is dependent on the value of the objective function of the solution represented by the length of the route, similarly to the approach discussed in the paper of Komaki, Sheikh, & Teymourian, 2014), or any unvisited node is drawn (when the list of vertices is empty). As a modification of the pitch – related to the *PAR* parameter – we opted for the choice (made within the nodes available) of the city nearest to the last visited site in the solution being created (the results of empirical studies assuming the alignment of the representation problem to continuous space and the use of the *bw* parameter showed the ineffectiveness of this approach in the case of ATSP).

In order to avoid premature convergence (the situation in which the algorithm gets stuck in the local optimum), we introduced the option of resetting the *HM* elements at the time of execution of a specified number of *R* iterations from the last replacement of the result in *HM*. This mechanism assumes that the best result is kept and the remaining solutions are drawn, forcing exploration of the space of solutions. Pseudocode of the proposed approach to the HS design is presented in Figure 2.

5. Empirical research methodology

The nineteen tasks representing the Asymmetric Traveling Salesman Problem were selected as the 'test bed'. Their characteristics are presented in Table 1. It is assumed that statistically significant results – for non-deterministic algorithms – can be achieved by repeating calculations at least ten times for each instance of the problem (Talbi, 2009), so each test was solved 30 times.

The values of the HS parameters were as follows: R=1000, HMS=5, $HMCR$=0.98, PAR=0.25 (empirical studies were carried out for their designation, the fragmented results of which are shown in Table 2). It was assumed that the algorithm would be executed for 10 minutes, during which the value of the objective function of the best solution found after 2, 6, and

The Harmony Search with different place of pitch adjustment pseudocode

```
1:  function HS(HMS, HMCR, PAR, IT, R, first city)
2:      iterations = 0
3:      iterationsFromTheLastReplacement = 0
4:      for i = 0; i < HMS; i++ do
5:          HM[i]=stochastically generate feasible solution
6:      end for
7:      Sort HM
8:      while iterations < IT do
9:          H[0]=first city
10:         for i = 1; i < n; i++ do                    ▷ n - number of cities
11:             Choose random r ∈ (0, 1)
12:             if r < HMCR then
13:                 list=generate list containing vertices occurring after H[i−1] in HM
14:                 if list.length > 0 then
15:                     H[i]=choose element ∈ list according to the roulette wheel
16:                 else
17:                     H[i]=choose randomly available city ∉ H
18:                 end if
19:                 Choose random k ∈ (0, 1)
20:                 if k < PAR then                     } Only HS
21:                     H[i]=find nearest and available city from H[i − 1]
22:                 end if
23:             else
24:                 H[i]=choose randomly available city ∉ H
25:                 Choose random k ∈ (0, 1)
26:                 if k < PAR then                     } Only HS′
27:                     H[i]=find nearest and available city from H[i − 1]
28:                 end if
29:             end if
30:             Choose random k ∈ (0, 1)
31:             if k < PAR then                         } Only HS″
32:                 H[i]=find nearest and available city from H[i − 1]
33:             end if
34:         end for
35:         if f(H) is better than f(HM[HMS − 1]) then
36:             HM[HMS − 1] = H
37:             Sort HM
38:             iterationsFromTheLastReplacement = 0
39:         else
40:             iterationsFromTheLastReplacement++
41:         end if
42:         if iterationsFromTheLastReplacement = R then
43:             for i = 1; i < HMS; i++ do
44:                 HM[i]=stochastically generate feasible solution
45:             end for
46:             Sort HM
47:             iterationsFromTheLastReplacement = 0
48:         end if
49:         iterations++
50:     end while
51:     return HM[0]
52: end function
```

Figure 3. The Harmony Search with different place of pitch adjustment pseudocode for ATSP.

Table 4. Compilation of the average error.

	Average error							
						HS		
Test name	NNA	GLS	HC	AMCPA (Osaba et al., 2014)	GA (Osaba et al., 2014)	2 minutes	6 minutes	10 minutes
br17	135.9	7.69	7.69	0.26	1.54	**0**	**0**	**0**
ftv33	30.87	23.64	23.64	7.77	7.79	2.47	**2.2**	**2.2**
ftv35	21.59	21.38	21.38	6.52	6.2	1.21	1.01	**0.94**
ftv38	16.21	10	10	5.33	6.92	1.39	0.94	**0.64**
p43	2.63	0.53	0.96	0.15	0.27	0.05	0.05	**0.03**
ftv44	24.86	24.18	24.18	10.49	8.38	2.05	1.63	**1.45**
ftv47	33.67	28.89	28.89	7.21	4.86	1.91	1.57	**1.44**
ry48p	16.19	15.21	13.81	2.79	4.67	1.11	0.8	**0.69**
ft53	37.78	30.93	30.14	12.64	11.98	10.39	7.65	**6.7**
ftv55	25.12	23.2	23.2	11.41	14.25	3.41	2.5	**1.98**
ftv64	43.5	36.54	36.22	13.18	14.86	3.57	2.4	**1.7**
ft70	11.67	8.15	8.98	4.4	5.45	4.49	4.04	**3.81**
ftv70	31.85	23.85	23.28	13.06	9.97	5.54	4.85	**4.23**
kro124p	31.12	26.82	24.77	**7.67**	10.58	11.07	8.94	8.12
ftv170	42.4	38.73	36.88	46.02	43.28	35.95	23.56	**21**
rbg323	30.77	**12.37**	12.67	43.4	60.11	60.61	57.7	56.71
rbg358	55.8	**17.02**	20.55	64.05	74.89	86.84	83.31	82
rbg403	43.41	6.98	**4.87**	17.52	20.83	32.67	31.67	31.05
rbg443	44.19	**7.21**	7.57	25.56	24.29	34.11	32.87	32.61
Average	35.77	19.12	18.93	15.76	17.43	15.73	14.09	**13.54**

10 minutes would be determined (the same time interval between measurements allows for the observation of changes in the dynamics of the improvement of the solution). Average error was calculated as follows: $((\text{average result} - \text{optimum})/\text{optimum}) \cdot 100\%$.

Algorithms were implemented in *C#* and the study was conducted on a Lenovo Y50-70 laptop, the parameters of which are presented in Table 3.

For comparative purposes – to provide background for the results generated by HS – the solutions obtained by the Nearest Neighbor Algorithm (NNA), Greedy Local Search (GLS) and Hill Climbing (HC) were selected. The first method always started constructing a route from city number one.

The Greedy Local Search variant is characterized with the acceptance of the first designated neighborhood solution, which is described with the more favorable value of the objective function, while Hill Climbing grants acceptance when the entire neighborhood is reviewed. The local search methods used in the study were based on the result found by the Nearest Neighbor Algorithm, and the neighborhood of the current solution was defined as a set of routes differing from it by two cities only, while the initial vertex remains unchanged.

The obtained results are also compared with the solutions presented in literature (Osaba et al., 2014). Due to the different formulation of the algorithm stop condition, they should not be treated as the basis for a direct comparison of the efficiency of metaheuristics, but should only be used to estimate the quality of the proposed solutions.

In order to determine the impact of the pitch adjustment operation on the effectiveness of the proposed HS design approach, an average error was determined for two additional HS variants: HS′ (in which the operation which depends on the probability *PAR* can only be executed when the criterion $r \geq HMCR$ is met) and HS″ (in which the pitch adjustment can occur regardless of the *HMCR* probability). To sum up, HS′ assumes that the greedy choice can be performed with $PAR \cdot (1 - HMCR)$ probability, while in HS″ this operation is

Table 5. Detailed compilation of the results for HS (objective function value).

				HS											
				2 minutes				6 minutes				10 minutes			
Test name	NNA	GLS	HC	Avg.	Min.	Max.	Sample st. dev.	Avg.	Min.	Max.	Sample st. dev.	Avg.	Min.	Max.	Sample st. dev.
br17	92	42	42	39	39	39	0	39	39	39	0	39	39	39	0
ftv33	1683	1590	1590	1317.73	1286	1355	27.12	1314.27	1286	1339	26.89	1314.27	1286	1339	26.89
ftv35	1791	1788	1788	1490.87	1473	1499	8.23	1487.87	1473	1499	10.01	1486.87	1473	1499	10.38
ftv38	1778	1683	1683	1551.23	1532	1580	11.69	1544.4	1532	1566	7.83	1539.77	1532	1549	7.34
p43	5768	5650	5674	5623.07	5620	5627	1.57	5622.53	5620	5627	1.78	5621.97	5620	5627	1.75
ftv44	2014	2003	2003	1646.13	1613	1728	25.75	1639.27	1613	1683	19.39	1636.4	1613	1683	18.27
ftv47	2374	2289	2289	1809.93	1776	1845	11.77	1803.9	1776	1814	11.28	1801.63	1776	1814	11.43
ry48p	16757	16615	16413	14582.27	14481	14921	111.41	14537.43	14459	14867	92.79	14522.07	14459	14707	51.86
ft53	9514	9041	8986	7622.1	7320	7989	193.33	7433.47	7177	7768	157.91	7367.6	7101	7666	137.33
ftv55	2012	1981	1981	1662.83	1608	1712	28.07	1648.13	1608	1699	27.65	1639.9	1608	1692	21.78
ftv64	2639	2511	2505	1904.67	1856	1963	29.56	1883.13	1850	1951	24.55	1870.30	1850	1901	17.03
ft70	43186	41824	42144	40411.1	39958	41042	223.5	40233.5	39567	40730	238.49	40146.03	39427	40586	250.41
ftv70	2571	2415	2404	2058	1973	2096	30.74	2044.57	1973	2096	35.5	2032.57	1973	2096	31.15
kro124p	47506	45947	45205	40239.1	39071	41305	588.51	39470.5	38240	40603	566.2	39171.53	38000	40330	579.56
ftv170	3923	3822	3771	3745.5	3270	4613	348.61	3404.17	3103	4038	192.18	3333.6	3101	4038	186.75
rbg323	1734	1490	1494	2129.7	2065	2208	39.99	2091.13	2007	2155	37.91	2077.93	2007	2146	33.51
rbg358	1812	1361	1402	2172.93	2068	2258	44.51	2131.9	2061	2215	40.5	2116.63	2044	2198	38.96
rbg403	3535	2637	2585	3270.43	3190	3323	29.63	3245.6	3185	3295	35.68	3230.37	3158	3292	30.57
rbg443	3922	2916	2926	3647.87	3576	3709	38.18	3613.93	3562	3664	25.92	3607.03	3562	3660	24

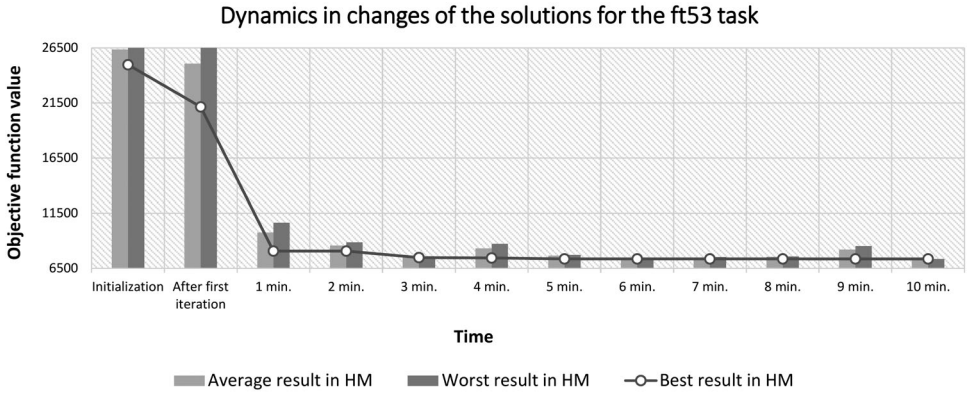

Figure 4. Dynamics in changes of the solutions for the ft53 task.

Table 6. HS convergence.

	HS convergence (no. of iterations)			
Test name	Avg.	Min.	Max.	Sample st. dev.
br17	645.53	26	6564	1249.51
ftv33	516091.17	10007	1560138	459200.14
ftv35	1439900.93	13218	6826948	1889012.96
ftv38	3548970.77	222341	7005398	2300263.01
p43	1490657.77	4908	5588569	1720942.9
ftv44	1562193.23	46083	4696309	1530379.93
ftv47	892287.57	53398	3893972	1097728.85
ry48p	1647047.73	161900	4753165	1198170.64
ft53	3760336.63	1456115	5313327	1117877.93
ftv55	1906755	176744	4140383	1250137.17
ftv64	1787198.8	302364	3267746	886740.57
ft70	2482584.27	473608	3998013	1068166.56
ftv70	1307286.17	52405	2677444	884120.36
kro124p	1259377.57	601975	2159931	427808.91
ftv170	556603.9	144731	817534	156975.58
rbg323	186538.29	16000	324302	87320.53
rbg358	181497.63	7420	328473	103979.73
rbg403	171730.62	6320	287079	86246.48
rbg443	96228.52	1352	183700	50670.06

executed with *PAR* probability (in the classic HS, the probability is equal to $PAR \cdot HMCR$). The pseudocode of the analyzed techniques is presented in Figure 3.

6. Results

The average error determined by the methods studied is shown in Table 4. The table also includes the reference results obtained with the Genetic Algorithm (GA) and the Adaptive Multi-Crossover Population Algorithm (AMCPA; results were processed based on Osaba et al., 2014).

Based on the results obtained, it was found that the proposed approach to the design of the HS algorithm is characterized with high efficacy. For most instances of the problem – the number of cities below 100 – the average percentage surplus of the objective function value, in relation to the optimum, was decidedly less than 5% (regardless of the time

Table 7. Compilation of the average error achieved by methods with different place of pitch adjustment operation.

	Average error								
	2 minutes			6 minutes			10 minutes		
Test name	HS	HS′	HS″	HS	HS′	HS″	HS	HS′	HS″
br17	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
ftv33	**2.47**	9.74	3.24	**2.2**	6.59	2.96	**2.2**	5.65	2.61
ftv35	**1.21**	6.92	1.39	**1.01**	4.4	1.28	**0.94**	3.87	1.13
ftv38	**1.39**	9.6	1.63	**0.94**	5.51	1.02	**0.64**	4.24	0.94
p43	**0.05**	0.21	**0.05**	0.05	0.16	**0.04**	**0.03**	0.13	**0.03**
ftv44	**2.05**	11.32	2.15	1.63	8.5	**1.62**	1.45	7.1	**1.08**
ftv47	**1.91**	9.82	1.99	**1.57**	6.12	1.7	**1.44**	4.84	1.61
ry48p	1.11	11.54	**1.07**	**0.8**	8.06	0.81	0.69	6.33	**0.65**
ft53	10.39	13.66	**10.08**	**7.65**	10.29	8.14	6.7	8.79	**6.5**
ftv55	3.41	11.71	**2.58**	2.5	8.82	**1.76**	1.98	7.01	**1.58**
ftv64	3.57	18.05	**3.55**	2.4	12.02	**2.16**	1.7	10.28	**1.68**
ft70	4.49	6.54	**4.36**	4.04	4.9	**3.94**	3.81	4.18	**3.78**
ftv70	5.54	20.07	**5.13**	4.85	13.68	**4.21**	4.23	11.71	**3.61**
kro124p	11.07	32.99	**9.43**	8.94	25.75	**7.8**	8.12	22.39	**7.04**
ftv170	35.95	110.79	**20.71**	23.56	82.52	**16.04**	21	72.17	**14.61**
rbg323	60.61	151.87	**56.24**	57.7	139.71	**53.22**	56.71	133.12	**51.41**
rbg358	86.84	225.91	**80.14**	83.31	211.61	**76.05**	82	202.86	**74.54**
rbg403	32.67	92.33	**30.76**	31.67	85.07	**29.44**	31.05	83.14	**29.03**
rbg443	34.11	96.29	**32.24**	32.87	91.04	**31.21**	32.61	89.16	**30.67**
Average	15.73	44.18	**14.04**	14.09	38.14	**12.81**	13.54	35.63	**12.24**

needed to apply the method). HS also rated the best results (among the analyzed algorithms; in terms of the objective function value) for 74% of the benchmarking tests.

For each of the analyzed time intervals in which the measurement was performed, HS obtained the most favorable results in terms of the average percentage surplus of the objective function values (within the methods under study), ranging from 15.73% to 13.54% (respectively for 2 and 10 minutes).

It is of particular interest that the efficiency of GLS and HC for tasks described by a minimum of 323 vertices is relatively high. Due to the multitude of permissible solutions, the proposed method, together with AMCPA and GA, yielded results with significantly higher values of the objective function.

Table 5 presents the values of the objective function determined by the tested solution methods. Accordingly, it was found that NNA, GLS and HC did not provide the optimal solution in any of the analyzed benchmarking tests. The proposed approach to the HS design made it possible to obtain the most favorable result for seven tasks, every time finding a given result in the first two minutes of running the algorithm. In time, one can observe a decrease in the dynamics of the improvement of the solution by HS (the visualization of the dependence is shown in Figure 4), but the process still produces the expected effects while avoiding stagnation – it would probably be possible to determine solutions optimal for each test by performing the method longer. In addition, the best results for the ftv38, ry48p, ftv64, and ftv70 tasks are characterized with the objective function value diverging from the optimum only to a small extent, which might demonstrate the need to improve the mechanism of exploiting the method (for example, by hybridizing it with other techniques).

Based on the above compilation, we argue that the algorithm is characterized by a significant standard deviation from the objective function value of the obtained solutions, thus

Table 8. Detailed compilation of the results for HS′ (objective function value).

	2 minutes				6 minutes				10 minutes			
Test name	Avg.	Min.	Max.	Sample st. dev.	Avg.	Min.	Max.	Sample st. dev.	Avg.	Min.	Max.	Sample st. dev.
br17	39	39	39	0	39	39	39	0	39	39	39	0
ftv33	1411.27	1316	1514	56.99	1370.73	1286	1472	48.04	1358.6	1286	1472	42.97
ftv35	1575	1473	1720	63.86	1537.8	1473	1632	43.68	1529.97	1473	1632	42.44
ftv38	1676.87	1540	1794	68.72	1614.33	1540	1733	48.48	1594.8	1540	1660	32.43
p43	5631.8	5622	5649	7.1	5629.07	5621	5646	5.98	5627.27	5620	5646	5.72
ftv44	1795.63	1672	1952	75	1750.07	1636	1915	56.55	1727.57	1636	1874	49.93
ftv47	1950.37	1814	2147	76.39	1884.7	1804	2056	54.21	1862	1803	1951	37.67
ry48p	16085.93	14867	16866	497.44	15583.9	14608	16460	447.1	15334.7	14607	16005	375.1
ft53	7848.2	7272	8779	348.54	7615.83	7046	8321	312.4	7512.27	6998	7934	248.16
ftv55	1796.37	1668	1920	60.37	1749.77	1609	1870	66.23	1720.7	1608	1870	56.6
ftv64	2170.97	2012	2333	90.12	2060.13	1881	2221	90.58	2028.03	1869	2220	88.85
ft70	41201.77	40262	42663	549.94	40567.7	39890	41276	360.41	40289.33	39420	40948	368.37
ftv70	2341.33	2183	2500	78.12	2216.73	2055	2454	80.91	2178.33	2041	2369	82.49
kro124p	48183.1	45068	52387	1566.79	45558.83	42735	47658	1156.95	44340.63	42349	46714	1118.78
ftv170	5807.13	5192	6520	315.31	5028.5	4579	5430	218.96	4743.4	4134	5094	231.6
rbg323	3339.73	3082	3534	88.85	3178.5	3001	3311	75.21	3091.2	2878	3215	79.19
rbg358	3790.3	3614	4008	83.54	3624.07	3350	3764	91.46	3522.3	3254	3685	92.74
rbg403	4740.87	4587	4849	63.71	4562.07	4465	4684	60	4514.33	4346	4620	56.65
rbg443	5339.1	5140	5496	74.8	5196.27	4995	5332	75.39	5145.2	4995	5291	73.69

Table 9. Detailed compilation of the results for HS′′ (objective function value).

	2 minutes				6 minutes				10 minutes			
Test name	Avg.	Min.	Max.	Sample st. dev.	Avg.	Min.	Max.	Sample st. dev.	Avg.	Min.	Max.	Sample st. dev.
br17	39	39	39	0	39	39	39	0	39	39	39	0
ftv33	1327.7	1286	1355	23.74	1324.1	1286	1339	23.74	1319.57	1286	1339	25.98
ftv35	1493.5	1473	1499	6.98	1491.8	1473	1499	8.73	1489.6	1473	1499	10.46
ftv38	1554.87	1532	1603	13.38	1545.63	1530	1549	5.32	1544.4	1530	1549	6.54
p43	5622.87	5620	5627	1.01	5622.4	5620	5627	1.48	5621.93	5620	5627	1.7
ftv44	1647.67	1613	1719	29.66	1639.1	1613	1683	23.13	1630.4	1613	1680	14.55
ftv47	1811.27	1776	1853	21.61	1806.27	1776	1846	19.36	1804.6	1776	1846	20.09
ry48p	14576.8	14507	14967	114.87	14538.47	14507	14914	82.82	14515.33	14507	14707	36.75
ft53	7601.37	7127	7985	232.25	7467.37	7083	7904	220.96	7354.13	7083	7809	185.27
ftv55	1649.53	1608	1716	34.18	1636.37	1608	1684	24.65	1633.33	1608	1680	22.6
ftv64	1904.37	1858	1980	28.77	1878.7	1842	1909	22.48	1869.87	1842	1909	22.03
ft70	40359.1	39469	40722	291.91	40197.83	39469	40567	258.07	40135	39444	40494	239.19
ftv70	2050.03	1979	2096	33.32	2032.03	1973	2096	28.54	2020.47	1955	2065	29.5
kro124p	39646.87	38136	40984	672.04	39054.37	37966	40920	723.2	38781.8	37625	40551	661.11
ftv170	3325.63	3063	3819	177.04	3197.03	3038	3643	127.92	3157.53	2977	3643	122.38
rbg323	2071.7	2015	2160	40.99	2031.73	1949	2104	33.07	2007.67	1888	2080	44.59
rbg358	2095.07	1969	2185	48.43	2047.47	1969	2135	46.95	2029.9	1933	2135	45.52
rbg403	3223.2	3172	3280	28.87	3190.6	3093	3268	36.58	3180.7	3093	3235	33.11
rbg443	3597	3513	3670	36.9	3568.9	3486	3621	35.2	3554.3	3486	3599	31.13

Table 10. Wilcoxon Signed-Rank Test results for the percentage surplus of the objective function value achieved by methods with different place of pitch adjustment operation.

*H*1*H*2	HS	HS′	HS″
HS	N/A	**4.92084E-88**	1
HS′	1	N/A	1
HS″	**2.66869E-38**	**1.16683E-88**	N/A

implying the observable non-determinism of the method leading to significantly different results after the same period of time (in consequence, preventing their prediction).

Table 6 shows the compilation of iterations after which the HS algorithm has converged. On its basis, it was found that in order to achieve stagnation (even for relatively small instances of the problem) it is necessary to perform a significant number of iterations.

Table 7 presents a compilation of the average error by HS, HS′ and HS″. Based on the results presented in Table 7, it was found that the lowest total percentage value surplus of the objective function in relation to the optimum is characterized by the variant HS″, whose significant efficiency was recorded for problem instances, which were described by above 43 vertices (which points to the effectiveness of the application heuristics of the Nearest Neighbor, indicating the direction of searching the space for solutions). Regardless of the time of performing the techniques, the reduction in the likelihood of making a pitch adjustment operation (the HS′ method) negatively affected the quality of the obtained results.

Table 8 and 9 show analyzed values of the objective function determined by the HS′ and the HS″.

To determine the recommended place of pitch adjustment operation, the Wilcoxon Signed-Rank Test was used for the percentage surplus of the objective function value, set by the analyzed HS variants after ten minutes. The value of 0.05 was adapted as a significance level (a lower result indicates that there are no grounds to undermine the alternative hypothesis according to which *H*1 yielded lower results than *H*2). The results are presented in Table 10. Based on their analysis, it is recommended that HS″ should be used, while the HS′ variant should be avoided.

7. Conclusions and future work

The obtained results indicate the relatively good effectiveness of the proposed approach (within the methods under study, it obtained the best – in terms of the total percentage surplus of the objective function value – results, characterized with the deviation from the optimum ranging from 13.54% to 15.73%, depending on the running time of the algorithm), encouraging further work on its refinement and the adaption of the method to solving other combinatorial problems.

Accounting for a variety of the obtained results close to the optimum and the observations discussed in the paper of Wang, Gao, and Zenger (2015), it is assumed that HS has a weak mechanism of exploitation (its inefficiency was revealed in particular for the tasks where the number of vertexes was above 300, for which the method had far worse results than the results obtained by simple local search methods that were based on the NNA), implying the need to conduct research on the hybridization of this method with other techniques to eliminate the imperfection.

The high effectivness of the HS″ variant achieved in the process of solving the instances of the problem that are described by a significant number of vertices and its lower effectiveness (compared to HS) for relatively small problems points to the potential appplicability of a dynamic value of the *PAR* parameter, which would combine the strengths of the two approaches. Regardless of the scale of the problem under analysis, we have noticed that the reduction in the likelihood of making a pitch adjustment operation (the HS′ method) negatively affected the quality of the obtained results.

Disclosure statement

No potential conflict of interest was reported by the authors.

Notes on contributors

Urszula Boryczka currently works at the Institute of Computer Science, University of Silesia in Katowice. She completed her Master's degree at University of Silesia in Sosnowiec in 1984 and she received Ph.D. degree at University in Wroclaw in 1993. She holds Doctor's of Science (Habilitation) degree in Computer Science from the Polish Academy of Science in Warsaw in 2009. Urszula Boryczka is a head of Division of Algorithmic and Computational Intelligence. She does research in algorithms and artificial intelligence, especially in computational swarm intelligence. Swarm intelligence is the discipline that deals with natural and artificial systems composed of many individuals that coordinate using decentralized control and self-organization. In particular, the discipline focuses on the collective behaviors that result from the local interactions of the individuals with each other and with their environment. Examples of systems studied by swarm intelligence are colonies of ants and bee colonies, flocks of birds and many others. ACO, BCO, PSO, CS and BA are examples of such systems examined in global or combinatorial optimization systems. Her current project is 'Ant colony algorithms in clustering problems'. Another issue is a 'Harmony search' which is a new metaheuristics in her study. This technique is now examined as a hybrid approach with other optimization techniques in aTSP problems.

Krzysztof Szwarc is currently a research assistant at the University of Silesia in Katowice. He received a B.Eng. degree in Logistics (2015) and an M.Eng. degree in Transport (2016) from the Silesian University of Technology. He obtained a B.Eng. (2017) and an M.Sc. (2018) degrees in Informatics from the University of Silesia in Katowice. His research interests focus on computational intelligence methods in transport, production and logistics.

References

Antosiewicz, M., Koloch, G., & Kamiński, B. (2013). Choice of best possible metaheuristic algorithm for the travelling salesman problem with limited computational time: Quality, uncertainty and speed. *Journal of Theoretical and Applied Computer Science*, *7*(1), 46–55.

Ayachi, I., Kammarti, R., Ksouri, M., & Borne, P. (2010). Harmony search algorithm for the container storage problem. *8th international conference of modeling and simulation – MOSIM'10*, Hammamet, Tunisia.

Bo, G., Huang, M., Ip, W. H., & Wang, X. (2009). The harmony search for the routing optimization in fourth party logistics with time windows. *2009 IEEE congress on evolutionary computation* (pp. 962–967). Trondheim.

Boryczka, U., & Szwarc, K. (2018). The adaptation of the harmony search algorithm to the ATSP. In N. Nguyen, D. Hoang, T. P. Hong, H. Pham, & B. Trawiński (Eds.), *Lecture notes in computer science: Vol. 10751. Intelligent information and database systems. ACIIDS 2018* (pp. 341–351). Cham: Springer.

Degertekin, S. O. (2008). Optimum design of steel frames using harmony search algorithm. *Structural and Multidisciplinary Optimization*, *36*(4), 393–401.

Forsati, R., Haghighat, A. T., & Mahdavi, M. (2008). Harmony search based algorithms for bandwidth-delay-constrained least-cost multicast routing. *Computer Communications*, *31*, 2505–2519.

Gao, K. Z., Suganthan, P. N., Pan, Q. K., Chua, T. J., Cai, T. X., & Chong, C. S. (2016). Discrete harmony search algorithm for flexible job shop scheduling problem with multiple objectives. *Journal of Intelligent Manufacturing*, *27*(2), 363–374.

Geem, Z., Kim, J., & Loganathan, G. (2001). A new heuristic optimization algorithm: Harmony search. *Simulation*, *76*(2), 60–68.

Hetmaniok, E., Jama, D., Słota, D., & Zielonka, A.(2011). Application of the harmony search algorithm in solving the inverse heat conduction problem. *Zeszyty Naukowe. Matematyka Stosowana, Wydawnictwo Politechniki Śląskiej, Zeszyt*, *1*, 99–108.

Komaki, M., Sheikh, S., & Teymourian, E. (2014). A hybrid harmony search algorithm to minimize total weighted tardiness in the permutation flow shop. *2014 IEEE symposium on computational intelligence in production and logistics systems* (pp. 1–8). Orlando, FL, USA.

Mrówczyńska, B., & Nowakowski, P. (2015). Optymalizacja tras przejazdu przy zbiórce zużytego sprzetu elektrycznego i elektronicznego dla zadanych lokalizacji punktów zbiórki [Optimization of collection routes of waste electrical and electronic equipment for the selected locations of collection points]. *Czasopismo Logistyka*, *2*, 593–604.

Nowakowski, P., Szwarc, K., & Boryczka, U. (2018). Vehicle route planning in e-waste mobile collection on demand supported by artificial intelligence algorithms. *Transportation Research Part D: Transport and Environment*, *63*, 1–22.

Öncan, T., Altınel, I. K., & Laporte, G. (2009). A comparative analysis of several asymmetric traveling salesman problem formulations. *Computers & Operations Research*, *36*(3), 637–654.

Osaba, E., Diaz, F., Onieva, E., Carballedo, R., & Perallos, A. (2014). A population metaheuristic with adaptive crossover probability and multi-crossover mechanism for solving combinatorial optimization problems. *International Journal of Artificial Intelligence*, *12*, 1–23.

Panchal, A. (2009). Harmony search in therapeutic medical physics. In Z. W. Geem (Ed.), *Studies in computational intelligence: Vol. 191. Music-inspired harmony search algorithm* (pp. 189–203). Berlin, Heidelberg: Springer.

Płaczek, E., & Szołtysek, J. (2008). Wybrane metody optymalizacji systemu transportu odpadów komunalnych w katowicach [Selected methods of municipal waste transport system optimization in katowice]. *LogForum*, *4*(2), 1–10.

Syberfeldt, A., Rogstrom, J., & Geertsen, A. (2015). Simulation-based optimization of a realworld travelling salesman problem using an evolutionary algorithm with a repair function. *International Journal of Artificial Intelligence and Expert Systems*, *6*(3), 27–39.

Talbi, E. (2009). *Metaheuristics: From design to implementation*. Hoboken, New Jersey: Wiley Publishing.

Wang, X., Gao, X., & Zenger, K. (2015). *An introduction to harmony search optimization method*. Heidelberg: Springer International Publishing.

Weyland, D. (2010). A rigorous analysis of the harmony search algorithm: How the research community can be misled by a 'Novel' methodology. *International Journal of Applied Metaheuristic Computing*, *1*(2), 50–60.